COURSE DESCRIPTION CARD - SYLLABUS

Course name
Object-oriented languages [S1MiKC1>JO]

Course

Field of study
Microelectronics and digital communications

Year/Semester
2/3

Area of study (specialization)
–

Profile of study
general academic

Level of study
first-cycle

Course offered in
Polish

Form of study
full-time

Requirements
compulsory

Number of hours

Lecture
24

Laboratory classes
30

Other
0

Tutorials
0

Projects/seminars
0

Number of credit points

3,00

Coordinators

dr inż. Marek Michalski
marek.michalski@put.poznan.pl

dr inż. Michał Sybis
michal.sybis@put.poznan.pl

Lecturers

Prerequisites

Basic knowledge of mathematical logic and combinatorics. The ability to create simple algorithms. Additionally, the student should have knowledge and basic programming skills, as well as proficiency in operating a PC. The ability to acquire information from literature and other sources in Polish or English, as well as to integrate, interpret, and draw conclusions from them. Awareness of the limitations of one's knowledge and skills, along with the need for continuous self-improvement and acquiring new competencies.

Course objective

The aim of the course is to introduce students to the fundamentals of software engineering, including the principles of object-oriented design. The classes cover key topics related to the practice of object- oriented programming in C++ and Java, encompassing both basic and more advanced language constructs.

Course-related learning outcomes

Knowledge:
1. He/she has advanced knowledge of the principles of computer program development and the use of learned C++/Java language structures

Skills:
1. He/she is able to develop simple programs using the C++/Java language to analyze and solve problems relevant to the field of study.
2. He/she is able to record, present, and process collected data in numerical and graphical form using the learned languages.
3. He/she is able to implement and utilize known mathematical models to solve problems using the C++/Java language.

Social competences:
1. He/she understands the necessity of expanding knowledge on the use of object-oriented programming.
2. He/she is aware of the possibilities and limitations of modern computer science while remaining open to its applications in new areas of daily life, the economy, technology, and science.
3. He/she has the ability to formulate personal opinions on currently used and available technologies and solutions in the design of modern IT systems.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Knowledge acquired in the lecture is verified by a colloquium or oral assessment carried out at the last lecture (typically). The assessment consists of 6 to 11 questions (multiple-choice and/or open-ended), which may have different point values. The passing threshold is 50% of the possible points.
Skills acquired in laboratory classes are verified on the basis of fulfilling tasks assigned in class or project. In both didactic forms, a passing threshold of 50% of the possible points is adopted. The following grading scale is used: < 50% 2.0; 50%-59% 3.0; 60%-69% 3.5; 70%-79% 4.0; 80%-89% 4.5; 90%-100% 5.0.

Programme content

The lecture focuses on object-oriented programming, covering key topics such as classes, objects, inheritance, polymorphism, and data encapsulation, which form the foundation of modern software design. Practical applications of the Standard Template Library (STL) will also be discussed, including data structures, algorithms, and iterators, which significantly facilitate working with C++ and Java. Additionally, issues related to secure programming will be addressed, including risks arising from suboptimal and incorrect code or vulnerabilities in selected solutions.

Course topics

During the C++ lectures and laboratory classes, the following topics are covered:
1. Classes and Objects (creation, attributes, constructors, destructors).
2. Memory Management and Pointers (operators new, delete, smart pointers).
3. Inheritance and Polymorphism (inheritance, multiple inheritance, virtual functions).
4. Exception Handling and Multithreading Design (try-catch mechanism, multi-thread handling, thread synchronization).
5. Advanced Object-Oriented Programming Techniques (templates, operator overloading).
6. Libraries (Standard Template Library - STL).
During the Java lectures and laboratory classes, the following topics are covered:
1. Basic Elements and Syntax of Java: data types, arithmetic operators and order of operations, input-output operations, loops, decision-making (conditional statements), functions, array-based operations.
2. Object-Oriented Concepts, including polymorphism, inheritance, different types of inheritance (abstract classes, interfaces), class definition packages, and scope of definitions.
3. Data Processing - reading and writing files in various formats, graphical presentation of results, selected libraries, database communication, and network communication. These topics will be discussed with a focus on secure implementation.
4. Implementation of Selected Probability Theory Concepts - generating pseudorandom numbers, calculating moments, elements of statistical analysis (distribution conformity, randomness).

5. Exception Handling and Multithreading Programming.
6. Implementation of Selected Network Services (e.g., HTTP).
7. Java-Specific Solutions, such as servlets and JSP (JavaServer Pages).

Teaching methods

Lecture: Multimedia presentation, illustrated with examples executed on a computer or presented on a whiteboard, possibly in a workshop format.
Laboratory classes: Performing tasks assigned by the instructor using computers - practical exercises, possibly supported by a multimedia presentation.

Bibliography

Basic:
Jerzy Grębosz, Opus Magnum C++11 : programowanie w języku C++. T. 1/T.2/T.3, Gliwice : Wydawnictwo Helion, 2020.
Bruce Eckel, Thinking in Java, Wydawnictwo Helion, 2006

Additional:
Kayshav Dattatri, Język C++. Efektywne programowanie obiektowe, Helion, 2005

Breakdown of average student's workload

	Hours	ECTS
Total workload	84	3,00
Classes requiring direct contact with the teacher	54	2,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	30	1,00